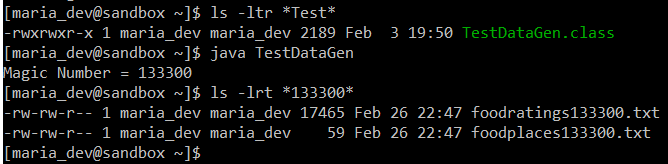
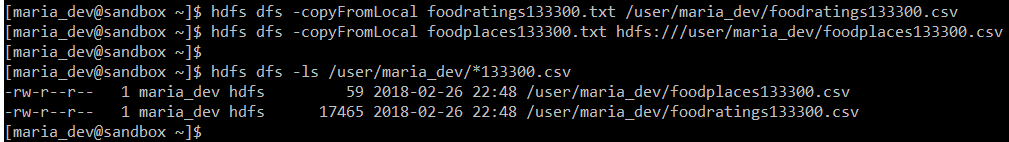
**CS595 - Assignment 7**

* Use the TestDataGen program from previous assignments to generate new data files

Copy the files to HDFS.



Magic Number = 133300



**Command Executed:**

java TestDataGen

hdfs dfs -copyFromLocal foodratings133300.txt /user/maria\_dev/foodratings133300.csv

hdfs dfs -copyFromLocal foodplaces133300.txt hdfs:///user/maria\_dev/foodplaces133300.csv

hdfs dfs -ls /user/maria\_dev/\*133300.csv

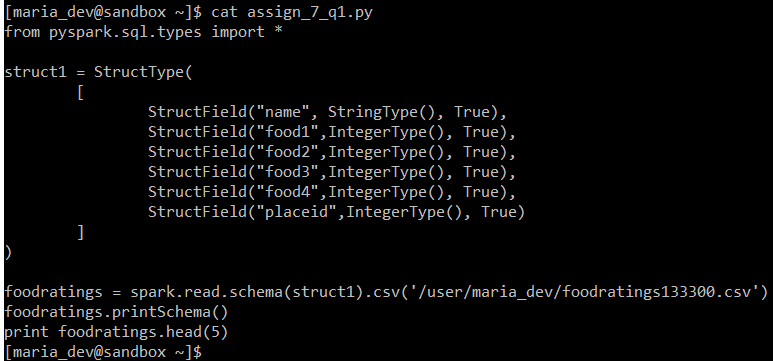
1. Load the ‘foodratings’ file as a ‘csv’ file into a DataFrame called ex1\_foodratings. When doing so specify a schema having fields of the following names and types:

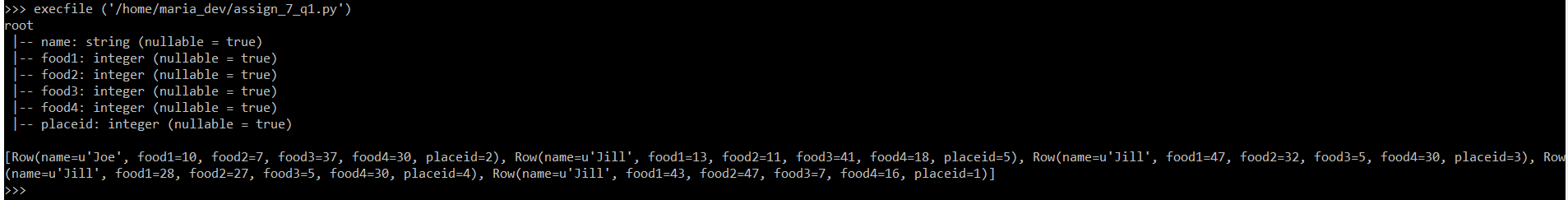
|  |  |
| --- | --- |
| Field Nampee | Field Type |
| Name | String |
| food1 | Integer |
| food1 | Integer |
| food1 | Integer |
| food1 | Integer |
| Placeid | Integer |

As the results of this exercise provide the magic number, the code you execute and screen shots of the following commands:

foodratings.printSchema()

foodratings.head(5)





**Command Executed:**

Magic Number = 133300

vi assign\_7\_q1.py

from pyspark.sql.types import \*

struct1 = StructType(

[

StructField("name", StringType(), True),

StructField("food1",IntegerType(), True),

StructField("food2",IntegerType(), True),

StructField("food3",IntegerType(), True),

StructField("food4",IntegerType(), True),

StructField("placeid",IntegerType(), True)

]

)

foodratings = spark.read.schema(struct1).csv('/user/maria\_dev/foodratings133300.csv')

foodratings.printSchema()

print foodratings.head(5)

:wq!

execfile ('/home/maria\_dev/assign\_7\_q1.py')

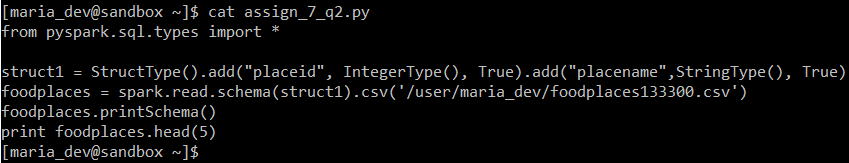
1. Load the ‘foodplaces’ file as a ‘csv’ file into a DataFrame called foodplaces. When doing so specify a schema having fields of the following names and types:

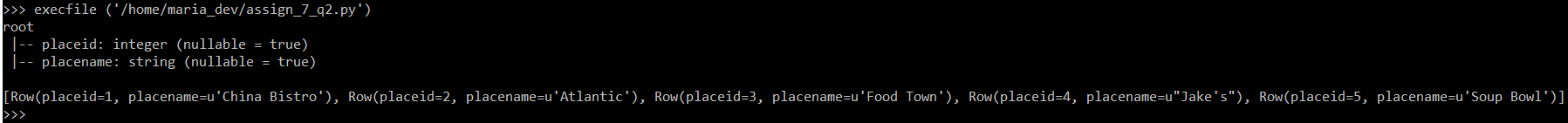
|  |  |
| --- | --- |
| Field Nampee | Field Type |
| Placeid | Integer |
| Placename | String |

As the results of this exercise provide the code you execute and screen shots of the following commands:

foodratings.printSchema()

foodratings.head(5)





**Command Executed:**

Magic Number = 133300

vi assign\_7\_q2.py

from pyspark.sql.types import \*

struct1 = StructType().add("placeid", IntegerType(), True).add("placename",StringType(), True)

foodplaces = spark.read.schema(struct1).csv('/user/maria\_dev/foodplaces133300.csv')

foodplaces.printSchema()

print foodplaces.head(5)

:wq!

execfile ('/home/maria\_dev/assign\_7\_q2.py')

1. **Step A**

Register the DataFrames created in exercise 1 and 2 as tables called “foodratingsT” and “foodplacesT”

**Step B**

Use a SQL query on the table “foodratingsT” to create a new DataFrame called foodratings\_ex3 holding records which meet the following condition: food2 < 25 and food4 > 40

As the results of this step provide the code you execute and screen shots of the following commands:

foodratings.printSchema()

foodratings.head(5)

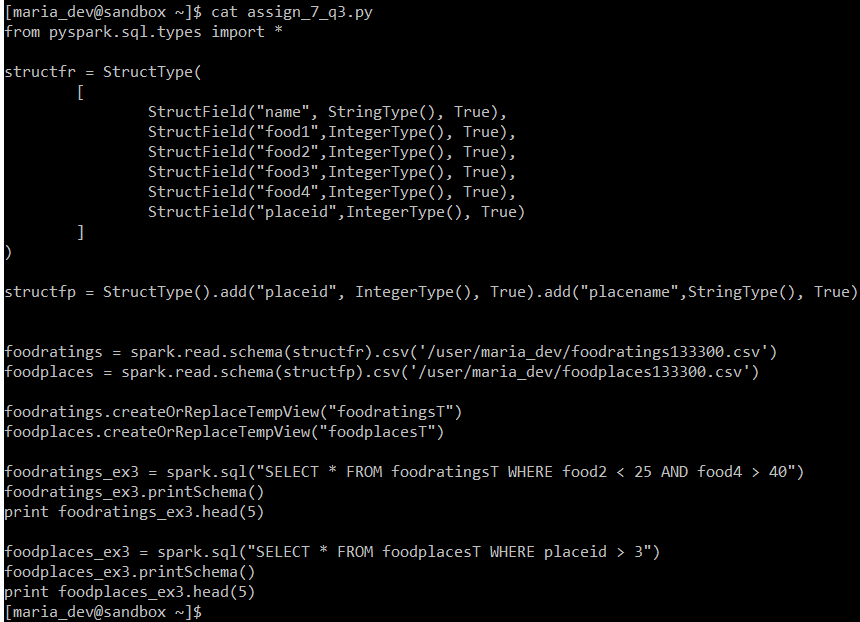
**Step C**

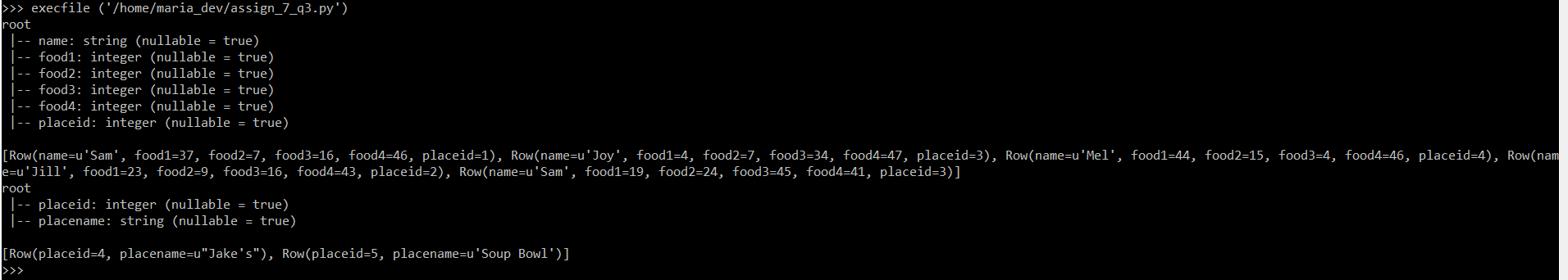
Use a SQL query on the table “foodplacesT” to create a new DataFrame called foodplaces\_ex3 holding records which meet the following condition: placeid > 3

As the results of this step provide the code you execute and screen shots of the following commands:

foodratings.printSchema()

foodratings.head(5)





**Command Executed:**

Magic Number = 133300

vi assign\_7\_q3.py

from pyspark.sql.types import \*

structfr = StructType(

[

StructField("name", StringType(), True),

StructField("food1",IntegerType(), True),

StructField("food2",IntegerType(), True),

StructField("food3",IntegerType(), True),

StructField("food4",IntegerType(), True),

StructField("placeid",IntegerType(), True)

]

)

structfp = StructType().add("placeid", IntegerType(), True).add("placename",StringType(), True)

foodratings = spark.read.schema(structfr).csv('/user/maria\_dev/foodratings133300.csv')

foodplaces = spark.read.schema(structfp).csv('/user/maria\_dev/foodplaces133300.csv')

foodratings.createOrReplaceTempView("foodratingsT")

foodplaces.createOrReplaceTempView("foodplacesT")

foodratings\_ex3 = spark.sql("SELECT \* FROM foodratingsT WHERE food2 < 25 AND food4 > 40")

foodratings\_ex3.printSchema()

print foodratings\_ex3.head(5)

foodplaces\_ex3 = spark.sql("SELECT \* FROM foodplacesT WHERE placeid > 3")

foodplaces\_ex3.printSchema()

print foodplaces\_ex3.head(5)

:wq!

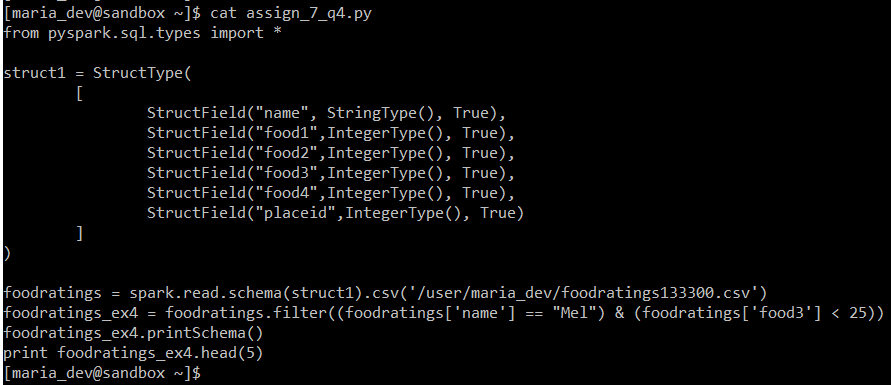
execfile ('/home/maria\_dev/assign\_7\_q3.py')

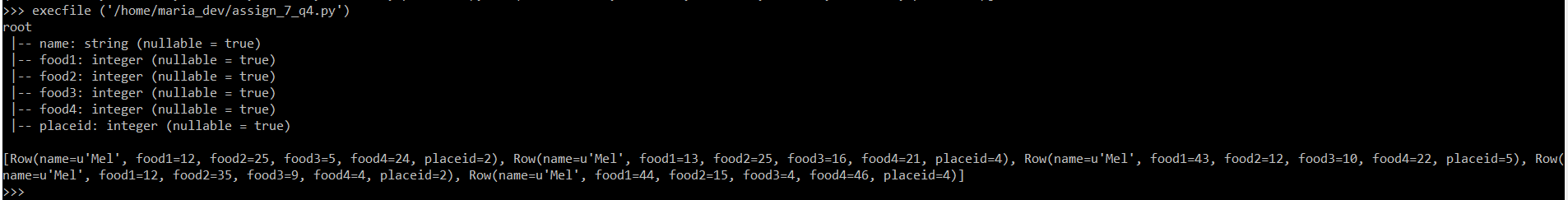
1. Use an operation (not a SQL query) on the DataFrame ‘foodratings’ create in exercise 1 to create a new DataFrame called foodratings\_ex4 that includes only those records (rows) where the ‘name’ field is “Mel” and food3 < 25.

As the results of this step provide the code you execute and screen shots of the following commands:

foodratings.printSchema()

foodratings.head(5)

****

****

**Command Executed:**

Magic Number = 133300

vi assign\_7\_q4.py

from pyspark.sql.types import \*

struct1 = StructType(

[

StructField("name", StringType(), True),

StructField("food1",IntegerType(), True),

StructField("food2",IntegerType(), True),

StructField("food3",IntegerType(), True),

StructField("food4",IntegerType(), True),

StructField("placeid",IntegerType(), True)

]

)

foodratings = spark.read.schema(struct1).csv('/user/maria\_dev/foodratings133300.csv')

foodratings\_ex4 = foodratings.filter((foodratings['name'] == "Mel") & (foodratings['food3'] < 25))

foodratings\_ex4.printSchema()

print foodratings\_ex4.head(5)

:wq!

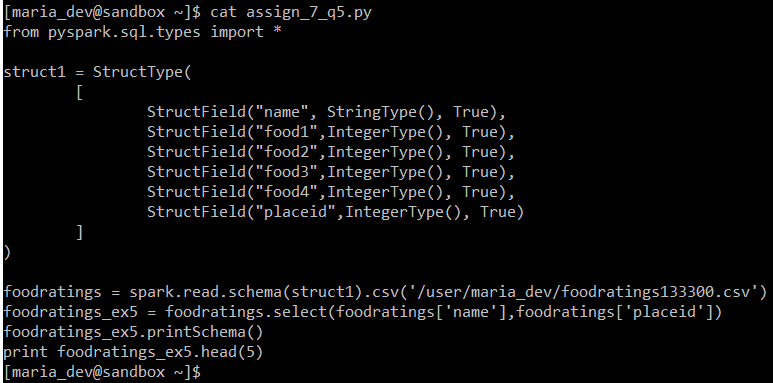
execfile ('/home/maria\_dev/assign\_7\_q4.py')

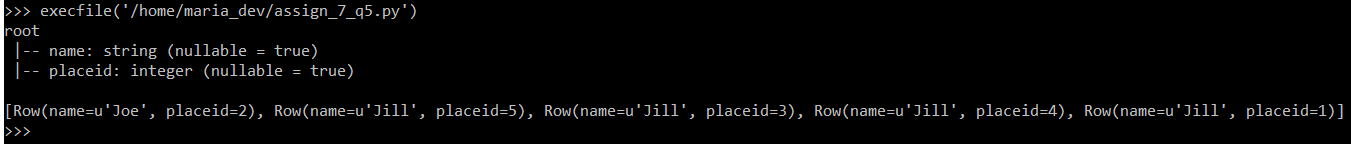
1. Use an operation (not a SQL query) on the DataFrame ‘foodratings’ create in exercise 1 to create a new DataFrame called foodratings\_ex5 that includes only the columns (fields) ‘name’ and ‘placeid’

As the results of this step provide the code you execute and screen shots of the following commands:

foodratings.printSchema()

foodratings.head(5)

****

****

**Command Executed:**

Magic Number = 133300

vi assign\_7\_q5.py

from pyspark.sql.types import \*

struct1 = StructType(

[

StructField("name", StringType(), True),

StructField("food1",IntegerType(), True),

StructField("food2",IntegerType(), True),

StructField("food3",IntegerType(), True),

StructField("food4",IntegerType(), True),

StructField("placeid",IntegerType(), True)

]

)

foodratings = spark.read.schema(struct1).csv('/user/maria\_dev/foodratings133300.csv')

foodratings\_ex5 = foodratings.select(foodratings['name'],foodratings['placeid'])

foodratings\_ex5.printSchema()

print foodratings\_ex5.head(5)

:wq!

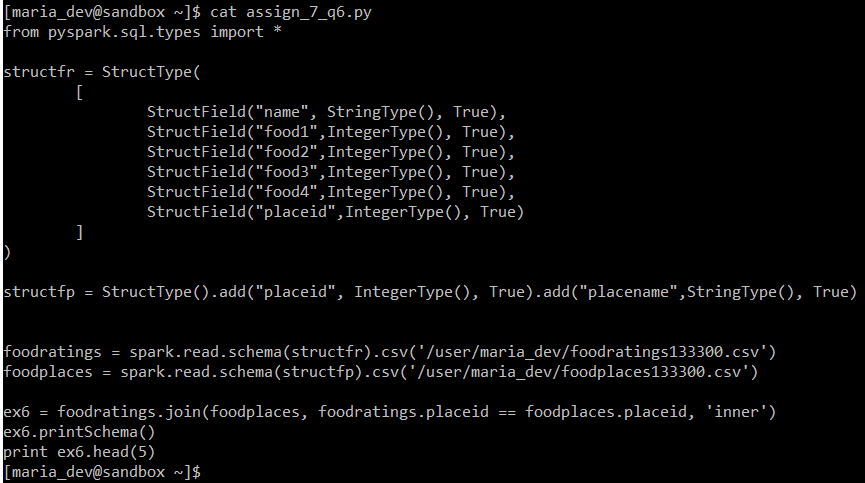
execfile ('/home/maria\_dev/assign\_7\_q5.py')

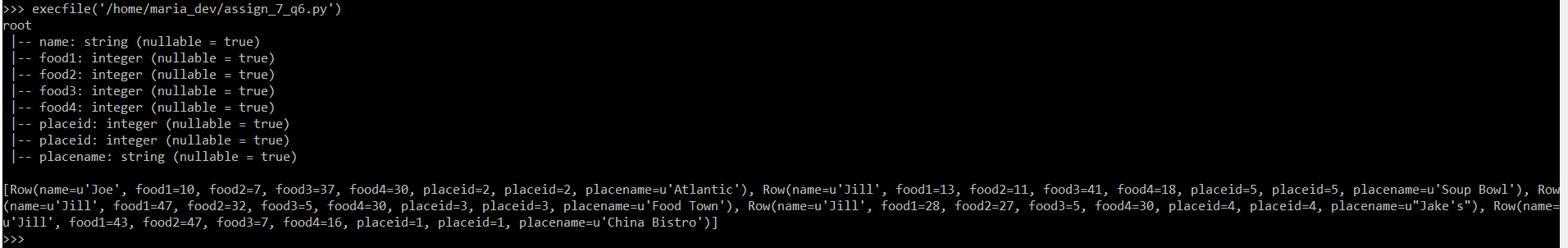
1. Use an operation on the DataFrame ‘to create a new DataFrame called ex6 which is the inner join, on placeid, of the DataFrames ‘foodratings; and ‘foodplaces’ created in exercises 1 and 2

As the results of this step provide the code you execute and screen shots of the following commands:

ex6.printSchema()

ex6.head(5)

****

****

**Command Executed:**

Magic Number = 133300

vi assign\_7\_q6.py

from pyspark.sql.types import \*

structfr = StructType(

[

StructField("name", StringType(), True),

StructField("food1",IntegerType(), True),

StructField("food2",IntegerType(), True),

StructField("food3",IntegerType(), True),

StructField("food4",IntegerType(), True),

StructField("placeid",IntegerType(), True)

]

)

structfp = StructType().add("placeid", IntegerType(), True).add("placename",StringType(), True)

foodratings = spark.read.schema(structfr).csv('/user/maria\_dev/foodratings133300.csv')

foodplaces = spark.read.schema(structfp).csv('/user/maria\_dev/foodplaces133300.csv')

ex6 = foodratings.join(foodplaces, foodratings.placeid == foodplaces.placeid, 'inner')

ex6.printSchema()

print ex6.head(5)

:wq!

execfile ('/home/maria\_dev/assign\_7\_q6.py')